Poster: Indoor Navigation for Visually Impaired People with Vertex Colored Graphs

Pei Du
Portland State University, pdu@pdx.edu

Nirupama Bulusu
Portland State University, nbulusu@pdx.edu

Follow this and additional works at: https://pdxscholar.library.pdx.edu/ece_fac

Let us know how access to this document benefits you.

Citation Details

This Article is brought to you for free and open access. It has been accepted for inclusion in Electrical and Computer Engineering Faculty Publications and Presentations by an authorized administrator of PDXScholar. Please contact us if we can make this document more accessible: pdxscholar@pdx.edu.
Poster: Indoor Navigation for Visually Impaired People with Vertex Colored Graphs

Pei Du
Portland State University
Portland, Oregon, USA
pdu@pdx.edu

Nirupama Bulusu
Portland State University
Portland, Oregon, USA
nbulusu@pdx.edu

ABSTRACT
Visually impaired people face many daily encumbrances. Traditional visual enhancements do not suffice to navigate indoor environments. In this paper, we explore path finding algorithms such as Dijkstra and A* combined with graph coloring to find a safest and shortest path for visual impaired people to navigate indoors. Our mobile application is based on a database which stores the locations of several spots in the building and their corresponding label. Visual impaired people select the start and destination when they want to find their way, and our mobile application will show the appropriate path which guarantees their safety.

CCS CONCEPTS
• Human-centered computing → Accessibility; Ubiquitous and mobile computing; Interaction design.

KEYWORDS
graph coloring, shortest path finding algorithm

ACM Reference Format:

1 INTRODUCTION
Visual impairment is a pervasive disability which consists of limited vision, blurry vision, light sensitivity and blind spots[1]. According to CDC, nearly 21 million people in United States have low vision problems, encountering daily encumbrances in activities such as indoor wayfinding, relying traditionally on audio and visual guidance [2–9]. However, these techniques do not always help them find the safest and shortest way to their destination. We build our wayfinding application based on graph theory [10] and our previous work [11]. A volunteer can use our system to automatically collect data for Assisted Navigation Systems, replacing human effort with deep learning, augmented reality and indoor positioning technologies.

Our application depends on the database that the volunteer collects. We consider the locations such as doors, elevators, stairs as graph nodes. Based on the labels in the database, we classify them as safe, cautious, and unsafe and mark them with different colors. And then we apply shortest path algorithms colored graphs to find the safest and shortest way for vision impaired people.

The contributions of our paper is to design an algorithm which uses Dijkstra and A* algorithms on colored graph to find the safest and shortest way for vision impaired people.

The contributions of our paper is to design an algorithm which uses Dijkstra and A* algorithms on colored graph to find the safest and shortest way for vision impaired people. And then we apply shortest path algorithms colored graphs to find the safest and shortest way for vision impaired people.

Figure 1: Graph Coloring (a) Basement (b) First floor

2 GRAPH COLORING
Graph Coloring is a step which needs to be accomplished before we run indoor wayfinding application. We use the floor plan of a specific building and consider it as a graph and classify the nodes as three categories, safe nodes, nodes with caution, and unsafe nodes.
The colors assigned to each category are green, orange and red respectively. For visually impaired people, it is very dangerous for them to use stairs to reach a different floor. Hence the stairs should be always be marked as red. In contrast, the elevators are very friendly to them, so we mark it as green. For some obstacles, such as desks which may have potential hazard or interrupt the user, we mark them as orange. If the user cannot find a path with green nodes, the application will show a path with a mixture of green and orange nodes, and the users should proceed with caution. In Figure 1, we assign a color to different spots and connect them with weighted edge based on the distance calculated from the positions.

We create two graphs $G_1, G_2$ to represent two floors and remove red nodes first to get $G_1r, G_2r$. Then, we also need to remove orange nodes $G_1o, G_2o$. In order to guarantee the safety of the user, $G_1o, G_2o$ have higher priority than $G_1r, G_2r$. Only when the algorithm cannot find a path in $G_1o, G_2o$, we need to choose a path with orange nodes and use $G_1r, G_2r$. For the floor the user starts their path, we find the shortest path from the start point to each elevator, save the path and total weights. Similarly, for the floor where the user’s destination lies, we find the shortest path from each elevator to the destination. We concatenate a final route with same connecting point (i.e. the elevator) and compare the total weights of each route and return the shortest route. Algorithm 1 shows the pseudo code of safest path finding algorithm. We test our algorithm on Android Studio, search our way from node 0 to node 9 and we attach the screenshot (see Figure 2) to show the safest and shortest path we find for the graph in Figure 1.

### Algorithm 1 Safest Path Finding Algorithm Between Two Floors

**Require:** $V_{\text{start}} \in G_1(E_1, V_1), V_{\text{end}} \in G_2(E_2, V_2)$

**Ensure:** $(V_{\text{start}}, V_{\text{elevator}}, \ldots, V_{\text{end}})$

1. Remove Red Nodes in $G_1, G_2$, get $G_1r, G_2r$
2. Remove Orange Nodes in $G_1r, G_2r$, get $G_1o, G_2o$
3. for $V_i \in V_{\text{elevator in } G_1o}$ do
4. Run Dijkstra or A* on $G_1o$ from $V_{\text{start}}$ to $V_i$
5. if Shortest path found then
6. save path
else
7. Run Dijkstra or A* on $G_1r$ from $V_{\text{start}}$ to $V_i$
8. save path
end if
end for

11. for $V_i \in V_{\text{elevator in } G_2o}$ do
12. Run Dijkstra or A* on $G_2o$ from $V_i$ to $V_{\text{end}}$
13. if Shortest path found then
14. save path
else
15. Run Dijkstra or A* on $G_2r$ from $V_i$ to $V_{\text{end}}$
16. save path
end if
end for
20. Concatenate path based on same $V_{\text{elevator}}$, and compare all the paths
return shortest path $(V_{\text{start}}, V_{\text{elevator}}, \ldots, V_{\text{end}})$

### 3 CONCLUSION

We have proposed an algorithm for visually impaired people to help them find their way safely and quickly in the indoor environment. Our method doesn’t need any installations for sensors or visible marks, only one volunteer can collect positions and labels of nodes by using automated AR-based annotation tool and then color the graph built from the floor plan. The users can hold a mobile device to guide their way to destination. We just finished the design of the algorithm and test it in the emulator. In the future, we have to calibrate the position and use AR tools or audio guidance to help them find their way in the real environment.

### REFERENCES